

# The Exponential Time Complexity of Computing the Probability That a Graph is Connected\*

Thore Husfeldt<sup>†</sup>      Nina Taslamán<sup>‡</sup>

## Abstract

We show that for every probability  $p$  with  $0 < p < 1$ , computation of all-terminal graph reliability with edge failure probability  $p$  requires time exponential in  $\Omega(m/\log^2 m)$  for simple graphs of  $m$  edges under the Exponential Time Hypothesis.

## 1 Introduction

Graph reliability is a simple mathematical model of connectedness in networks that are subject to random failure of its communication channels. This type of stochastic networks arise naturally in, e.g., communication or traffic control; see [1] for an extensive survey of application areas.

For a connected graph  $G = (V, E)$  and probability  $p$ , the *all-terminal reliability*  $R(G; p)$  is the probability that there is a path of operational edges between every pair of nodes, given that every edge of the graph fails independently with probability  $p$ . For example, with  $p = \frac{1}{2}$ , the all-terminal reliability of the graph  $\mathfrak{C}_5$  is  $\frac{3}{16}$ , and the all-terminal reliability of the graph  $\mathfrak{K}_5$  is  $\frac{2}{16}$ .

In general, for a connected, undirected graph  $G = (V, E)$  with edge-failure probability  $p$ , the all-terminal reliability can be given as

$$R(G; p) = \sum_{\substack{A \subseteq E \\ \text{spanning} \\ \text{connected}}} p^{|E \setminus A|} (1 - p)^{|A|}. \quad (1)$$

For example  $R(\mathfrak{C}_5; \frac{1}{3}) = (\frac{2}{3})^5 + 5 \cdot \frac{1}{3} \cdot (\frac{2}{3})^4 = \frac{112}{243}$ . Computing  $R(G; p)$  directly from (1) can take up to  $O(2^m)$  operations, where  $m$  is the number of edges in  $G$ . An algorithm of Buzacott [4] solves the problem in time  $3^n n^{O(1)}$  for graphs of  $n$  vertices. Further improvements exist [2], but remain exponential in  $n$ ; this is explained by the  $\exp(\Omega(n))$  lower bound of [5]. On the other hand, subexponential time algorithms have been found for some restricted classes of graphs. For example, the problem can be solved in time  $\exp(O(\sqrt{n}))$  for planar graphs [14]. A natural question is then whether the complexity can be reduced for further classes of graphs. Especially, from an applications point of view, the case of simple graphs is interesting.

\*To appear in *5th International Symposium on Parameterized and Exact Computation (IPEC 2010)*, December 13-15, 2010, Chennai, India, Springer LNCS, 2010. Partially supported by Swedish Research Council grant VR 2007–6595.

<sup>†</sup>IT University of Copenhagen, Denmark and Lund University, Sweden

<sup>‡</sup>IT University of Copenhagen, Denmark

It is clear that  $R(G; 0) = 1$  and  $R(G; 1) = 0$  for all connected graphs, so for some values of  $p$  the problem is trivial. One may ask what the situation is for values close to these extremes. Moreover, for  $p = \frac{1}{2}$  it is easily seen that (1) equals the number of connected, spanning subgraphs of  $G$ , divided by  $2^m$ . This is an interesting enumeration problem in itself, and one could be tempted to hope for a better algorithm than  $\exp(O(n))$ , because a related enumeration problem, the number of spanning *trees* of  $G$ , can be solved in polynomial time by Kirchhoff's matrix-tree theorem [11].

## Result

We give a lower bound on the problem of computing all-terminal graph reliability for the class of simple graphs for all nontrivial  $p$ , in the framework recently proposed by Dell *et al.* [5]. In particular, we work under the counting exponential time hypothesis:

(#ETH) There is a constant  $c > 0$  such that no deterministic algorithm can compute #3-Sat in time  $\exp(cn)$ .

This is a relaxation of the exponential time hypothesis (ETH) of Impagliazzo *et al.* [9], so our results hold under ETH as well. The best current bound for #3-Sat is  $O(1.6423^n)$  [12].

**Theorem 1.** *For any fixed probability  $p$  with  $0 < p < 1$ , computing the all-terminal reliability  $R(G; p)$  of a given simple graph  $G$  of  $m$  edges requires time exponential in  $\Omega(m/\log^2 m)$  under #ETH.*

In particular, the bound holds for  $p = \frac{1}{2}$ , i.e., counting the number of connected spanning subgraphs of a given graph.

We have expressed the lower bound in terms of the parameter  $m$ , the number of edges of the input graph. Since  $n \leq m$  for connected graphs, the result implies the lower bound  $\exp(\Omega(n/\log^2 n))$  in terms of the parameter  $n$ , the number of vertices of the input graph. Moreover, the  $\Omega(m/\log^2 m)$  lower bound together with the  $\exp(O(n))$  algorithm from [4, 2] shows that the hard instances have roughly linear density, ruling out a better algorithm than  $\exp(O(n/\log^2 n))$  also for the restricted case of sparse graphs.

Our bound does not quite match the best known upper bound  $\exp(O(n))$  of [4, 2]. This situation is similar to the bounds reported in [5] for related problems on simple graphs, which also fall a few logarithmic factors (in the exponent) short of the best known algorithms. The bound does, however, suffice to separate the complexity of reliability computation from the  $\exp(O(\sqrt{n}))$  bound for the planar case [14].

## Graph polynomials

Expression (1), viewed as a function  $p \mapsto R(G; p)$  for fixed  $G$ , is known as the *reliability polynomial* of  $G$ , an object studied in algebraic graph theory [6, Sec. 15.8]. For example,  $R(\mathfrak{C}_5; p) = (1 - p)^5 + 5p(1 - p)^4$ .

Arguably, the most important graph polynomial is the bivariate *Tutte polynomial*  $T(G; x, y)$ , which encodes numerous combinatorial parameters of the input graph  $G$ , and whose restriction to certain lines and curves in the  $xy$ -plane

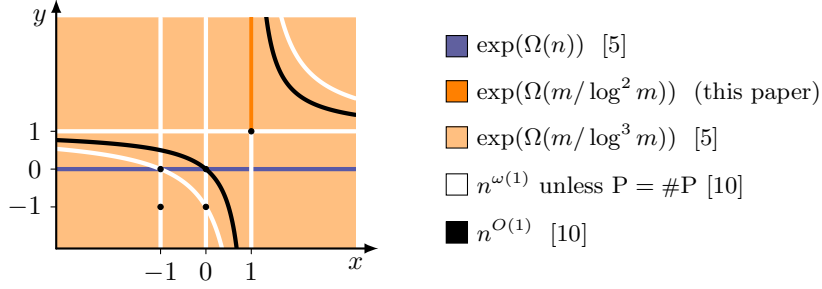


Figure 1: Exponential time complexity under  $\#ETH$  of the Tutte plane for simple graphs.

specialize to other well known graph polynomials. The reliability polynomial is essentially a restriction of this polynomial to the ray  $\{(1, y) : y > 1\}$ . The complexity of computing  $T(G; x, y)$  at various points  $(x, y)$  with respect to an input graph  $G$  is very well-studied in various models of computation, and the present paper thus establishes lower bounds for simple graphs along the mentioned ray, which was left open in a recent study [5] to completely map the exponential time complexity of the “Tutte plane”.

## Related work

The structural complexity of all-terminal graph reliability was studied by Provan and Ball [13]. For any probability  $p$ , with  $0 < p < 1$ , it is shown that computing  $R(G; p)$  for given  $G$  is hard for Valiant’s counting class  $\#P$  [16]. The reductions in [13] do not preserve the parameters  $n$  and  $m$ , so that the running time bounds under  $\#ETH$  implicitly provided by their techniques are typically exponential in  $\Omega(n^{1/k})$  for some  $k$ .

The reliability problem under consideration in this paper admits a number of natural extensions.

1. We can consider the computational problem of finding the reliability polynomial itself, instead of its value at a fixed point  $p$ . The input to this problem is a graph, and the output is a list of coefficients. For example, on input  $\textcircled{3}$  the output should give  $R(\textcircled{3}; p) = 4p^5 - 15p^4 + 20p^3 - 10p^2 + 1$ .
2. We can associate individual probabilities to every edge. For example, the graph  $\textcircled{\frac{1}{2} \quad \frac{1}{4}}$  becomes disconnected with probability  $\frac{5}{8}$ .
3. We can consider multigraphs like  $\textcircled{\infty}$ , but with the same edge weight  $p$ . As indicated by the examples (for  $p = \frac{1}{2}$ ), the multigraph case is a special case of the individually edge-weighted case, a fact that we will use later.

All of these problems are at least as hard as the problem under consideration in the present paper. Lower bounds of size  $\exp(\Omega(m))$  are given in [5] or follow relatively easily (see §2.3).

A recent paper of Hoffman [8] studies the complexity of another graph polynomial, the independent set polynomial, in the same framework.

## 2 Preliminaries

We will only be concerned with undirected graphs. For a graph  $G = (V, E)$  let  $n$  denote the number of vertices,  $m$  the number of edges, and for any subset  $A \subseteq E$  let  $\kappa(A)$  denote the number of connected components in the subgraph  $(V, A)$  (especially,  $\kappa(A) = 1$  means that the edge subset  $A$  is spanning and connected). Also, for graph polynomials  $P$  and  $Q$ , we write  $P(G; \mathbf{x}) \sim Q(G'; \mathbf{x}')$  if the two expressions are equal up to an easily computable factor.

### 2.1 Weighted reliability

The reliability polynomial can be formulated as a restriction of the *Tutte polynomial*, which for an undirected graph  $G$  is given by

$$T(G; x, y) = \sum_{A \subseteq E} (x-1)^{\kappa(A)-\kappa(E)} (y-1)^{\kappa(A)+|A|-|V|}.$$

Note that  $\kappa(E) = 1$  in our case. We find the reliability polynomial along the ray  $\{(1, y) : y > 1\}$  in the so called *Tutte plane*, as  $R(G; p) \sim T(G; 1, 1/p)$ ; in full detail:

$$R(G; p) = p^{m-n+1} (1-p)^{n-1} T(G; 1, 1/p), \quad (0 < p < 1). \quad (2)$$

For complexity analysis of the Tutte polynomial, it has proved a considerable technical simplification to consider Sokal's *multivariate Tutte polynomial* [15]. Here the graph is equipped with some weight function  $\mathbf{w} : E \rightarrow \mathbb{R}$ , and the polynomial is given by

$$Z(G; q, \mathbf{w}) = \sum_{A \subseteq E} \mathbf{w}(A) q^{\kappa(A)},$$

where  $\mathbf{w}(A) = \prod_{e \in A} \mathbf{w}(e)$  is the edge-weight product of the subset  $A$ . For constant edge weights  $w = y - 1$  we have  $Z(G; q, w) \sim T(G; x, y)$  with  $q = (x-1)(y-1)$ . The “reliability line”  $x = 1$  in the Tutte plane thus corresponds to  $q = 0$  in the weighted setting, where  $Z$  vanishes, so instead we will consider the slightly modified polynomial

$$\hat{Z}(G; q, \mathbf{w}) = q^{-1} Z(G; q, \mathbf{w}).$$

At  $q = 0$ , this gives a weighted version of the reliability polynomial:

**Definition 1.** (*Weighted reliability polynomial*)

For a connected, undirected graph  $G = (V, E)$ , the *weighted reliability polynomial* of  $G$  is given by

$$\hat{R}(G; \mathbf{w}) = \hat{Z}(G; q, \mathbf{w})|_{q=0} = \sum_{\substack{A \subseteq E \\ \kappa(A)=1}} \mathbf{w}(A). \quad (3)$$

For constant edge weight  $w > 0$  we have  $\hat{R}(G; w) = w^{n-1} T(G; 1, 1+w)$ , so for  $0 < p < 1$  we can recover the reliability polynomial through (2) as

$$R(G; p) = p^m \hat{R}(G; 1/p - 1). \quad (4)$$

## 2.2 Graph transformations

A classical technique for investigating the complexity of the Tutte polynomial at a certain point  $(x', y')$  of the Tutte plane, is to relate it to some already settled point  $(x, y)$  via a graph transformation  $\varphi$ , such that  $T(G; x', y') \sim T(\varphi(G); x, y)$ . For the weighted setting we have the following rules, which are simple generalizations of [7, Sec. 4.3]. (See Appendix A.1.) For a graph  $G = (V, E)$  with edge weights given by  $\mathbf{w}$ :

**Lemma 1.** *If  $\varphi(G)$  is obtained from  $G$  by replacing a single edge  $e \in E$  with a simple path of  $k$  edges  $P = \{e_1, \dots, e_k\}$  with  $\mathbf{w}(e_i) = w_i$ , then*

$$\hat{R}(\varphi(G); \mathbf{w}) = C_P \cdot \hat{R}(G; \mathbf{w}[e \mapsto w']),$$

where

$$\frac{1}{w'} = \frac{1}{w_1} + \dots + \frac{1}{w_k} \quad \text{and} \quad C_P = \frac{1}{w'} \prod_{i=1}^k w_i.$$

**Lemma 2.** *If  $\varphi(G)$  is obtained from  $G$  by replacing a single edge  $e \in E$  with a bundle of parallel edges  $B = \{e_1, \dots, e_k\}$  with  $\mathbf{w}(e_i) = w_i$ , then*

$$\hat{R}(\varphi(G); \mathbf{w}) = \hat{R}(G; \mathbf{w}[e \mapsto w']),$$

where

$$w' = -1 + \prod_{i=1}^k (1 + w_i).$$

**Corollary 1.** *If  $\varphi(G)$  is obtained from  $G$  by replacing a single edge  $e \in E$  with a simple path of  $k$  edges of constant weight  $w$ , then*

$$\hat{R}(\varphi(G); \mathbf{w}) = kw^{k-1} \cdot \hat{R}(G; \mathbf{w}[e \mapsto w/k]), \quad (5)$$

and if it is obtained from  $G$  by replacing  $e \in E$  with a bundle of  $k$  parallel edges of constant weight  $w$ , then

$$\hat{R}(\varphi(G); \mathbf{w}) = \hat{R}(G; \mathbf{w}[e \mapsto (1 + w)^k - 1]). \quad (6)$$

These rules are transitive [7, Lem. 1], and so can be freely combined for more intricate weight shifts. To preserve constant weight functions we need to perform the same transformation to every edge of the graph. This calls for the graph theoretic version of Brylawski's tensor product for matroids [3]. We found the following terminology more intuitive for our setting:

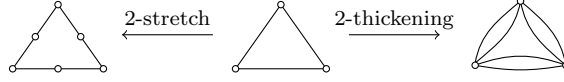
**Definition 2.** (*Graph inflation*)

Let  $H$  be a 2-terminal undirected graph. For any undirected graph  $G = (V, E)$ , an  $H$ -inflation of  $G$ , denoted  $G \otimes H$ , is obtained by replacing every edge  $xy \in E$  by (a fresh copy of)  $H$ , identifying  $x$  with one of the terminals of  $H$  and  $y$  with the other.<sup>1</sup>

If  $H$  is a simple path of  $k$  edges,  $G \otimes H$  gives the  $k$ -stretch of  $G$ . Similarly, a bundle of  $k$  parallel edges results in a  $k$ -thickening, of  $G$ .

---

<sup>1</sup> This can, in general, be done in two different ways, resulting in graphs that need not be isomorphic. However, the Tutte polynomial is blind to this difference. See extensive footnote in [5], Section 5.1.



### 2.3 Hardness of computing coefficients

Our pivot for proving Theorem 1 will be the following hardness result, which says that even when restricted to fixed hyperbolas  $(x-1)(y-1) = q$ , computing the full Tutte polynomial is hard. This is an extension to the case  $q = 0$  of Lemma 2 in [5], and the proof is given in Appendix 2.

**Lemma 3.** *Under #ETH, computing the coefficients of the polynomial  $w \mapsto \hat{Z}(G; q, w)$  for given simple graph  $G$  and rational number  $q \notin \{1, 2\}$  requires time exponential in  $\Omega(m)$ .*

Since  $R(G; p)$  is essentially  $\hat{R}(G; \mathbf{w})$  restricted to positive constant weight functions, and since  $\hat{R}(G; \mathbf{w}) = \hat{Z}(G; 0, \mathbf{w})$ , the following is immediate:

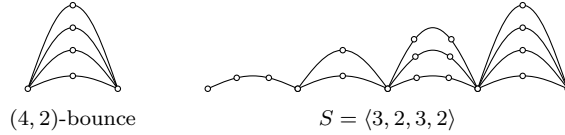
**Corollary 2.** *Under #ETH, it requires time exponential in  $\Omega(m)$  to compute the coefficients of the reliability polynomial.*

## 3 Bounce graphs

As a first step towards Theorem 1, we present here a class of graph transformations whose corresponding weight shifts for the reliability polynomial are all distinct. These transformations are mildly inspired by  $k$ -byte numbers, in the sense that each has associated to it a sequence of length  $k$ , such that the lexicographic order of these sequences determines the numerical order of the corresponding (shifted) weights. Each transformation is a *bounce inflation*:

**Definition 3.** (*Bounce graph*)

For positive numbers  $h$  (height) and  $l$  (length), the  $(h, l)$ -*bounce* is the graph obtained by identifying all the left and all the right endpoints of  $h$  simple paths of length  $l$ . Given a *bounce sequence*,  $S = \langle s_1, s_2, \dots, s_k \rangle$ , of  $k$  numbers  $s_i > 1$ , the corresponding *bounce graph*,  $B_S$ , is the (simple) graph obtained by concatenating  $k$   $(h, l)$ -bounces by their endpoints, where the height starts at 1 for the first bounce and then increases by one for each follower, and the length of the  $i$ th bounce is  $s_i$ .



The *length* of a bounce graph is the number of bounces in it (or, equivalently, the height of the highest bounce).

Inflation by a bounce graph has the following weight-shifting effect, from  $\hat{R}$ 's perspective:

**Lemma 4.** *For a graph  $G$ , bounce sequence  $S = \langle s_1, s_2, \dots, s_k \rangle$  and  $w > 0$ :*

$$\hat{R}(G \otimes B_S; w) = C_S^m \cdot \hat{R}(G; w_S),$$

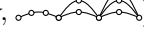

where

$$\frac{1}{w_S} = \sum_{i=1}^k \frac{1}{(1 + w/s_i)^i - 1} \quad \text{and} \quad C_S = \frac{1}{w_S} \cdot \prod_{i=1}^k w^{(s_i-1)i} ((w + s_i)^i - s_i^i).$$

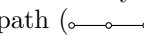
*Proof.* Starting out with  $G \otimes B_S$ , we will look at the effect of replacing one of the  $m$  copies of  $B_S$  with a single edge  $e$ . We show that, with  $\varphi$  denoting this operation:

$$\hat{R}(G \otimes B_S; w) = C_S \cdot \hat{R}(\varphi(G \otimes B_S); \mathbf{w}[e \mapsto w_S]), \quad (7)$$

where  $w_S$  has the above form, and  $\mathbf{w}$  has the old value  $w$  on all unaffected edges. The lemma follows from performing  $\varphi$  for every copy of  $B_S$  in  $G \otimes B_S$ .

The first step towards transforming a bounce graph (say, ) into a single edge, is to replace each path of each bounce in it by a single edge. Applying (5) of Corollary 1 to each path of the  $i$ th bounce gives a factor  $(s_i w^{s_i-1})^i$  to the polynomial, and each edge in the resulting  $(h, 1)$ -bounce gets weight  $w/s_i$  in the modified graph. Repeating this process for every bounce gives a simplified bounce graph () in a transformed graph  $\phi(G \otimes B_S)$  such that

$$\hat{R}(G \otimes B_S; w) = \left( \prod_{i=1}^k (s_i w^{s_i-1})^i \right) \cdot \hat{R}(\phi(G \otimes B_S); \mathbf{w}'),$$

with  $\mathbf{w}'$  taking the value  $w/s_i$  for every edge in the  $i$ th bounce of the simplified bounce graph, and the old value  $w$  outside it. Next, we successively replace each of its  $(h, 1)$ -bounces by a single edge, to get a simple path () of length  $k$  (with non-constant edge weights). From (6) of Corollary 1, we know that this does not produce any new factors for the polynomial, but the weight of the  $i$ th edge in this path will be given by

$$w_i = (1 + w/s_i)^i - 1.$$

Finally, we compress the path into a single edge  $e$ . A single application of Lemma 1 then gives the result in (7).  $\square$

If Lemma 3 is our pivot for proving Theorem 1, the following result is the lever:

**Lemma 5.** *For any size  $m$ , there exist  $m+1$  distinct, simple bounce graphs  $B_S$  of size  $O(\log^2 m)$ , such that for any two associated bounce sequences  $S$  and  $T$ :*

$$S >_{lex} T \Rightarrow w_S < w_T \quad (8)$$

for all  $w > 6$ .

*Proof.* The set of bounce sequences  $S = \langle s_1, \dots, s_l \rangle$  of length  $l = \log(m+1)$  and with each  $s_i \in \{2, 3\}$ , provides  $m+1$  different simple bounce graphs of promised size. To show that any two of them satisfy equation (8) we will look at the difference

$$\Delta_{S,T}(w) = \frac{1}{w_S} - \frac{1}{w_T},$$

and show that  $\Delta_{S,T}(w) > 0$  for  $w > 6$ .

Let  $k$  be the first index where the sequences differ, say  $s_k = 3$  and  $t_k = 2$ . We then have

$$\begin{aligned}\Delta_{S,T}(w) &= \frac{1}{(1+w/3)^k - 1} + \sum_{i=k+1}^l \frac{1}{(1+w/s_i)^i - 1} \\ &\quad - \frac{1}{(1+w/2)^k - 1} - \sum_{i=k+1}^l \frac{1}{(1+w/t_k)^i - 1}.\end{aligned}$$

This would be minimal if  $s_i = 2$  and  $t_i = 3$  for all  $i > k$ , i.e. if

$$\Delta_{S,T}(w) = f(1+w/3) - f(1+w/2),$$

where

$$f(x) = \frac{1}{x^k - 1} - \sum_{i=k+1}^l \frac{1}{x^i - 1}.$$

If we could show that  $f'(x) < 0$  for  $x > x_0$  then it would follow (e.g. from the mean value theorem) that  $f(x) > f(y)$  for  $x < y$  above  $x_0$ . In particular, with  $x_0 = 3$  this would prove our claim. To see that this is indeed the case, we look at the derivative

$$f'(x) = -\frac{kx^{k-1}}{(x^k - 1)^2} + \sum_{i=k+1}^l \frac{ix^{i-1}}{(x^i - 1)^2}.$$

A bit of manipulation shows that the terms of the sum, let us call them  $T_i$ , satisfy  $T_i > 2T_{i+1}$  for  $x > 3$ , so

$$f'(x) < \frac{kx^{k-1}}{(x^k - 1)^2} \left( -1 + \sum_{i=k+1}^l \frac{1}{2^i} \right) < 0$$

for  $x > 3$ , and we are done by the above argument.  $\square$

## 4 Evaluating the reliability polynomial is hard

We are ready to prove Theorem 1. We introduce the following notation for the problem of evaluating a graph polynomial  $P(G; \mathbf{x})$  at a given point  $\mathbf{x}$

$P\text{-VAL}(\mathbf{x})$ :

**input**    A simple, connected, undirected graph  $G$   
**output**    The value of  $P(G; \mathbf{x})$  (a rational number)

In this notation, the computational problem described in Theorem 1 can be written as  $R\text{-VAL}(p)$ , and the corresponding problem for weighted reliability with constant edge weight  $w$  is  $\hat{R}\text{-VAL}(w)$ . We will prove Theorem 1 by reducing the problem of computing coefficients of the polynomial  $p \mapsto R(G; p)$ , to the problem  $R\text{-VAL}(p)$  for any arbitrary fixed probability  $p$  with  $0 < p < 1$ .



*Proof (of Theorem 1).* Let  $G$  be a simple graph with  $n$  vertices and  $m$  edges. We prove that  $\hat{R}\text{-VAL}(w)$  requires time exponential in  $\Omega(m/\log^2 m)$  for any  $w > 0$ , which by (4) gives the same bound for  $R\text{-VAL}(p)$  for any  $p$  with  $0 < p < 1$ .

Suppose we have an algorithm for  $\hat{R}\text{-VAL}(w)$  for some fixed  $w > 0$ . For simplicity of exposition, first assume  $w > 6$ . From Lemma 5 we can easily construct  $m + 1$  bounce graphs  $B_S$  such that each  $\hat{R}(G \otimes B_S; w)$  gives us the value of  $\hat{R}(G; w_S)$  at some new weight  $w_S$ , with all such  $w_S$  distinct. Computing  $\hat{R}\text{-VAL}(w)$  for each of these  $m + 1$  bounce inflations, we get the value of  $\hat{R}(G; w)$  at  $m + 1$  distinct  $w$ -values. Since the degree of this polynomial is  $m$ , that gives us the coefficients by interpolation. By Corollary 2, the whole process must then require time  $\exp(\Omega(m))$ . By Lemma 5, each inflation  $G \otimes B_S$  will have  $O(m \log^2 m)$  edges. Thus, for graphs of size  $O(m \log^2 m)$ , the problem  $\hat{R}\text{-VAL}(w)$  has a lower bound of  $\exp(\Omega(m))$ . The claimed bound then follows from the fact that

$$\varphi(m) = m \log^2 m \implies \varphi^{-1}(m) \in O(m/\log^2 m) .$$

We turn to the case  $0 < w \leq 6$ . Given such a  $w$ , choose a number  $k$  such that

$$w' := (w/2 + 1)^k - 1 > 6 .$$

From the above, we know that for any simple graph  $G$  it takes time exponential in  $\Omega(m/\log^2 m)$  to evaluate  $\hat{R}(G; w')$ . Now consider the  $k$ -thickening-2-stretch of  $G$ , let us call this  $G'$ . This will be a simple graph of  $O(km)$  edges, and from Corollary 1 it follows that the value of  $\hat{R}(G'; w)$  would give us the value of  $\hat{R}(G; w')$ . Thus, computing the former must also require time exponential in  $\Omega(m/\log^2 m)$ . Since  $G'$  has  $O(km)$  edges, this gives a lower bound for  $\hat{R}\text{-VAL}(w)$  exponential in  $\Omega(m/(k \log^2 m))$ , which is  $\Omega(m/\log^2 m)$  as a function of  $m$ .  $\square$

## 5 Remarks

For the multivariate Tutte polynomial  $Z(G; q, w)$ , the current lower bound for simple graphs is  $\exp(\Omega(m/\log^3 m))$  [5]. One might ask whether the bounce graph construction could be used to improve this. The weight shift corresponding to a bounce inflation is in this case given by (for  $w \neq 0$  and  $q \notin \{0, -2w\}$ )

$$\frac{q}{w_S} = \prod_{i=1}^k \left( \frac{q}{\{(q/r_i) + 1\}^i - 1} - 1 \right) , \quad \text{where } r_i = \left(1 + \frac{q}{w}\right)^{s_i} - 1$$

Unlike the expression in Lemma 4 this is not a sum of powers, so the ‘ $k$ -byte number’ analogy is remote, and there is now also a dependency on  $q$  which seems to make it difficult proving something like Lemma 5 under the same constraints.

## References

- [1] M. O. Ball, C. J. Colbourn, and J. S. Provan. *Network Reliability*, volume 7 of *Handbooks in operations research and management science*, chapter 11. Elsevier Science, 1995.

- [2] A. Björklund, T. Husfeldt, P. Kaski, and M. Koivisto. Computing the Tutte polynomial in vertex-exponential time. In *49th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2008), October 25–28, 2008, Philadelphia, Pennsylvania, USA*, pages 677–686. IEEE Computer Society, 2008.
- [3] T. H. Brylawski. The Tutte polynomial. In *Matroid Theory and Its Applications*, pages 125–275. C.I.M.E., Ed. Liguori, Napoli & Birkhser, 1980.
- [4] J. A. Buzacott. A recursive algorithm for finding reliability measures related to the connection of nodes in a graph. *Networks*, 10(4):311–327, 1980.
- [5] H. Dell, T. Husfeldt, and M. Wahlén. Exponential time complexity of the permanent and the Tutte polynomial. In *37th International Colloquium on Automata, Languages and Programming (ICALP 2010), July 5–10, 2010, Bordeaux, France*, volume 6198 of *LNCS*, pages 426–437. Springer, 2010. Full paper in Electronic Colloquium on Computational Complexity, Report No. 78 (2010).
- [6] C. Godsil and G. Royle. *Algebraic Graph Theory*. Graduate Texts in Mathematics. Springer, 2001.
- [7] L. A. Goldberg and M. Jerrum. Inapproximability of the Tutte polynomial. *Inform. Comput.*, 206(7):908–929, 2008.
- [8] C. Hoffmann. Exponential time complexity of weighted counting of independent sets. In *5th International Symposium on Parameterized and Exact Computation (IPEC 2010), December 13–15, 2010, Chennai, India*, *LNCS*. Springer, 2010.
- [9] R. Impagliazzo, R. Paturi, and F. Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001.
- [10] F. Jaeger, D. L. Vertigan, and D. J. Welsh. On the computational complexity of the Jones and Tutte polynomials. *Math. Proc. Cambridge*, 108(1):35–53, 1990.
- [11] G. Kirchhoff. Über die Auflösung der Gleichungen, auf welche man bei der Untersuchung der linearen Verteilung galvanischer Ströme geführt wird. *Ann. Phys. Chem.*, 72:497–508, 1847.
- [12] K. Kutzkov. New upper bound for the #3-SAT problem. *Inf. Process. Lett.*, 105(1):1–5, 2007.
- [13] J. S. Provan and M. O. Ball. The complexity of counting cuts and of computing the probability that a graph is connected. *SIAM J. Comput.*, 12(4):777–788, 1983.
- [14] K. Sekine, H. Imai, and S. Tani. Computing the Tutte polynomial of a graph of moderate size. In *6th International Symposium of Algorithms and Computation (ISAAC 1995), December 4–6, 1995, Cairns, Australia*, volume 1004 of *LNCS*, pages 224–233. Springer, 1995.

- [15] A. D. Sokal. The multivariate Tutte polynomial (alias Potts model) for graphs and matroids. In *Surveys in Combinatorics*, pages 173–226. Cambridge University Press, 2005.
- [16] L. G. Valiant. The complexity of enumeration and reliability problems. *SIAM J. Comput.*, 8(3):410–421, 1979.

## A Supplementary proofs

### A.1 Lemma 1 and 2

*Proof (of Lemma 1 and 2).* We repeat the arguments from the proof in [10, Sec.4.3]: Let  $S$  be the set of subsets  $A \subseteq E \setminus \{e\}$  that already span the whole graph  $G$ , i.e.

$$S = \{A \subseteq E \setminus \{e\} : \kappa(A) = 1\},$$

and let  $T$  be the set of subsets that need the edge  $e$  to span the graph:

$$T = \{A \subseteq E \setminus \{e\} : \kappa(A) = 2 \text{ and } \kappa(A \cup \{e\}) = 1\}.$$

With  $w'$  denoting the weight of the edge  $e$  in the original graph  $G$ , (3) gives, for both lemmas,

$$\hat{R}(G; \mathbf{w}[e \mapsto w']) = \sum_{A \in S} \mathbf{w}(A)(1 + w') + \sum_{A \in T} \mathbf{w}(A)w'. \quad (9)$$

We will compare the partial sums here to the corresponding ones obtained when we alter the graph. When  $\varphi$  is the operation described in Lemma 1, we have (with  $P$  the set of edges in the path)

$$\begin{aligned} \hat{R}(\varphi(G); \mathbf{w}) &= \sum_{A \in S} \mathbf{w}(A) \left( \mathbf{w}(P) + \sum_{i=1}^k \mathbf{w}(P \setminus e_i) \right) + \sum_{A \in T} \mathbf{w}(A) \mathbf{w}(P) = \\ &= \sum_{A \in S} \mathbf{w}(A) \left( \prod_{i=1}^k w_i + \sum_{j=1}^k \prod_{i \neq j} w_i \right) + \sum_{A \in T} \mathbf{w}(A) \prod_{i=1}^k w_i. \end{aligned}$$

Comparing corresponding sums to (9), it is easy to check that the expressions for  $w'$  and  $C_P$  in Lemma 1 indeed make  $\hat{R}(\varphi(G); \mathbf{w}) = C_P \cdot \hat{R}(G; \mathbf{w}[e \mapsto w'])$ .

When  $\varphi$  is the operation described in Lemma 2, we have (with  $B$  the set of edges in the bundle)

$$\hat{R}(\varphi(G); \mathbf{w}) = \sum_{A \in S} \mathbf{w}(A) \left( 1 + \sum_{\substack{A' \subseteq B \\ A' \neq \emptyset}} \mathbf{w}(A') \right) + \sum_{A \in T} \mathbf{w}(A) \left( \sum_{\substack{A' \subseteq B \\ A' \neq \emptyset}} \mathbf{w}(A') \right),$$

and Lemma 2 follows since

$$\sum_{\substack{A' \subseteq B \\ A' \neq \emptyset}} \mathbf{w}(A') = \prod_{i=1}^k (w_i + 1) - 1.$$

□

## A.2 Deletion/contraction and Lemma 3

For an edge  $e = xy \in E$ , let  $G \setminus e$  be the graph obtained by deleting  $e$ , and let  $G/e$  be the (multi)graph obtained by *contracting*  $e$ , i.e. by identifying the end vertices  $x$  and  $y$  before removing  $e$ . With  $w_e$  denoting the weight of edge  $e$ , we have the following *deletion/contraction reduction* for the weighted Tutte polynomial (see [15, Sec. 4.3])

$$Z(G; q, \mathbf{w}) = Z(G \setminus e; q, \mathbf{w}) + w_e \cdot Z(G/e; q, \mathbf{w}).$$

Note that if  $e \in E$  is a bridge, then  $G \setminus e$  has one more component than  $G$ , while in any other case both  $G \setminus e$  and  $G/e$  have the same number of connected components as  $G$ . Using the above identity and the fact that  $\hat{Z}(G; q, \mathbf{w}) = q^{-1}Z(G; q, \mathbf{w})$ , this gives:

$$\hat{Z}(G; q, \mathbf{w}) = \begin{cases} q \cdot \hat{Z}(G \setminus e; q, \mathbf{w}) + w_e \cdot \hat{Z}(G/e; q, \mathbf{w}) & \text{if } e \text{ is a bridge,} \\ \hat{Z}(G \setminus e; q, \mathbf{w}) + w_e \cdot \hat{Z}(G/e; q, \mathbf{w}) & \text{otherwise.} \end{cases} \quad (10)$$

*Proof (of Lemma 3).* The case  $q \neq 0$  is treated in Lemma 2 of [5]. For  $q = 0$ , we give a reduction to Lemma 1 from the same paper. Under our assumptions it says that  $w \mapsto \hat{Z}(G; 0, \mathbf{w})$  cannot be computed faster than  $\exp(\Omega(m))$ , where  $\mathbf{w}$  is given by (for some set  $T$  of three edges):

$$\mathbf{w}(e) = \begin{cases} -1, & \text{if } e \in T, \\ w, & \text{otherwise.} \end{cases} \quad (11)$$

The proof of this actually uses the restriction that  $G' = (V, E \setminus T)$  is connected, so we can assume that this is the case. Thus, no edge in  $T$  is a bridge. Three applications of (10), to delete/contract these edges, gives

$$\hat{Z}(G; 0, \mathbf{w}) = \sum_{C \subseteq \{1,2,3\}} (-1)^{|C|} \hat{Z}(G_C; 0, w),$$

for some graphs  $G_C$  of constant edge-weight  $w$ . These  $G_C$ 's may contain loops and multiple edges from the contractions. To address this we look at the 2-stretch  $G_C \otimes P_3$  of each  $G_C$ . This will give simple graphs of constant weight functions, and  $m$  applications of (5) from Corollary 1 gives

$$\hat{Z}(G_C \otimes P_3; 0, w) = (2w)^m \hat{Z}(G_C; 0, w/2).$$

If an algorithm could compute the coefficients of  $w \mapsto \hat{Z}(G; 0, w)$  faster than  $\exp(\Omega(m))$  for any simple graph  $G$ , it could be used to compute eight polynomials  $w \mapsto \hat{Z}(G_C \otimes P_3; 0, w)$  (one for each subset  $C$ ). This would give us first  $w \mapsto \hat{Z}(G_C; 0, w)$  and then  $w \mapsto \hat{Z}(G; 0, \mathbf{w})$ , faster than  $\exp(\Omega(m))$ , which is impossible according to Lemma 1 from [5].  $\square$